

# Parallel Analysis and Visualization on Cray Linux Environment

**David Pugmire**

Oak Ridge National Laboratory

**Hank Childs**

Lawrence Livermore National Laboratory

**Sean Ahern**

Oak Ridge National Laboratory

# Overview

## Challenges of analysis at the petascale

### Scale

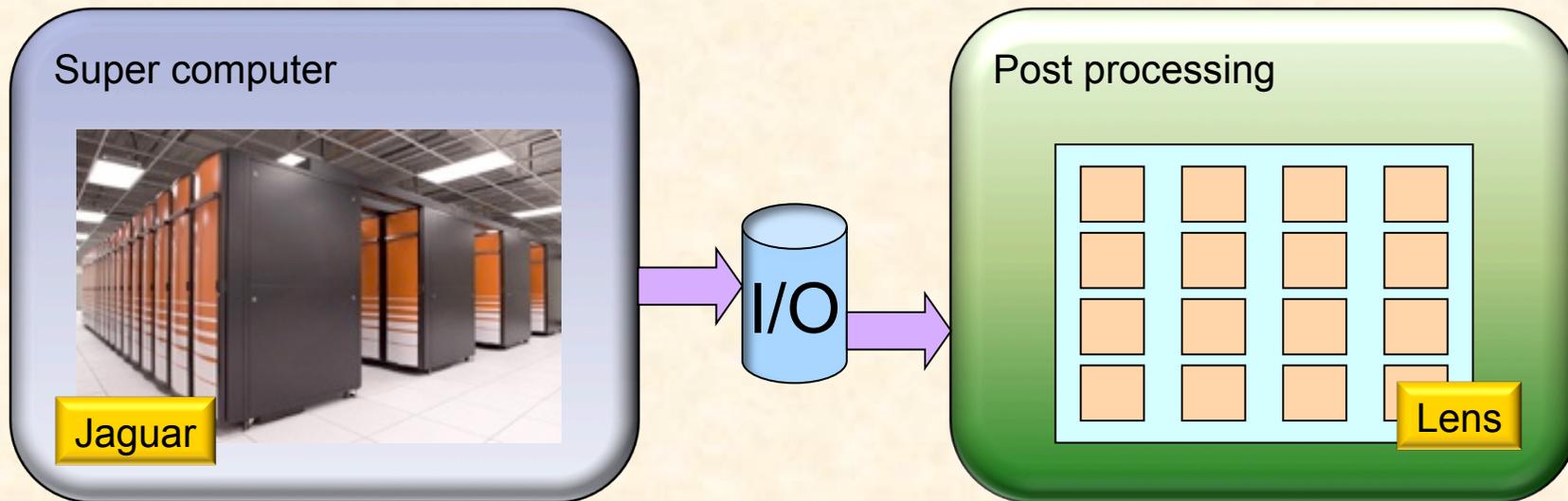
- Size
- Not compute bound
- Viz is always IO bound

### Complexity

- More science, not (always) more resolution.
- Requires solutions **beyond** pure parallelism.
- New paradigms needed

# Production analysis and viz

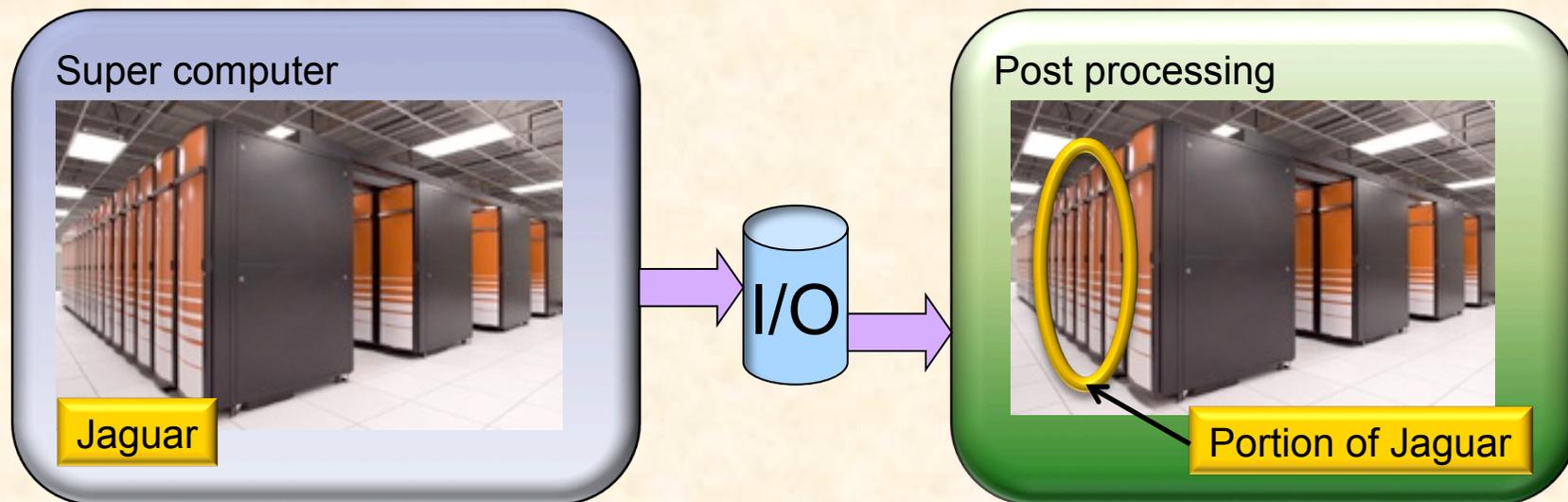
- Method 1: Traditional Visualization



- Shared disk (hopefully)
- Graphics cards

# Production analysis and viz

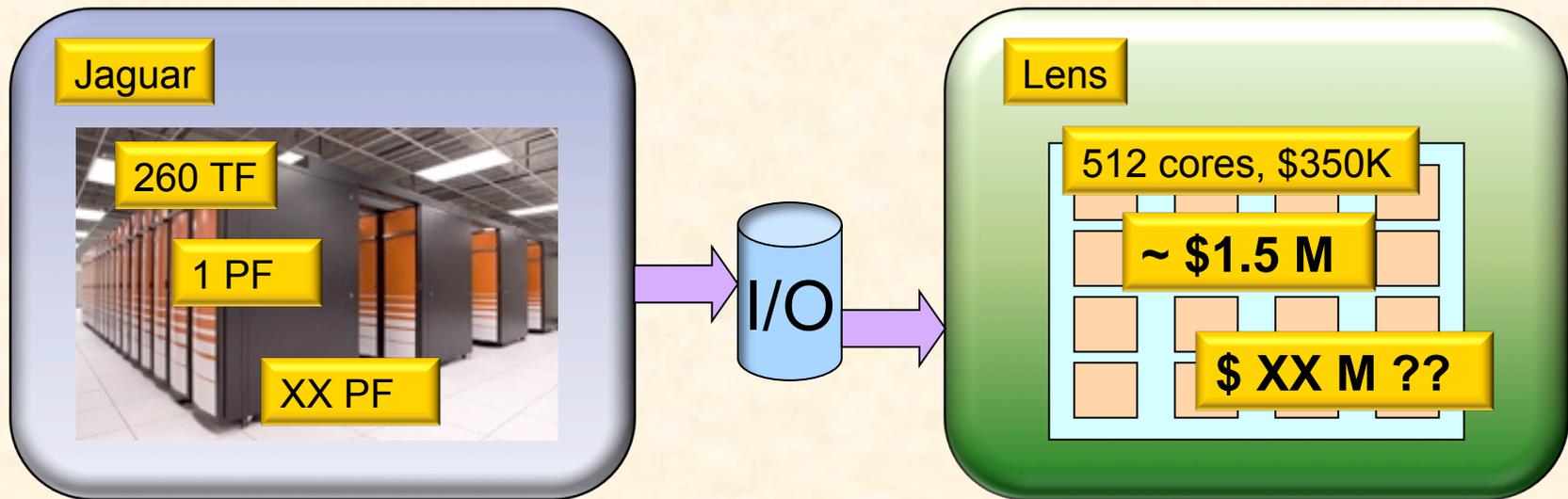
- Method 2: Shared resource visualization



- Computing, post processing both done on SC
- Simulation writes to disk, post processing reads from disk
- No graphics hardware

# Production analysis and viz

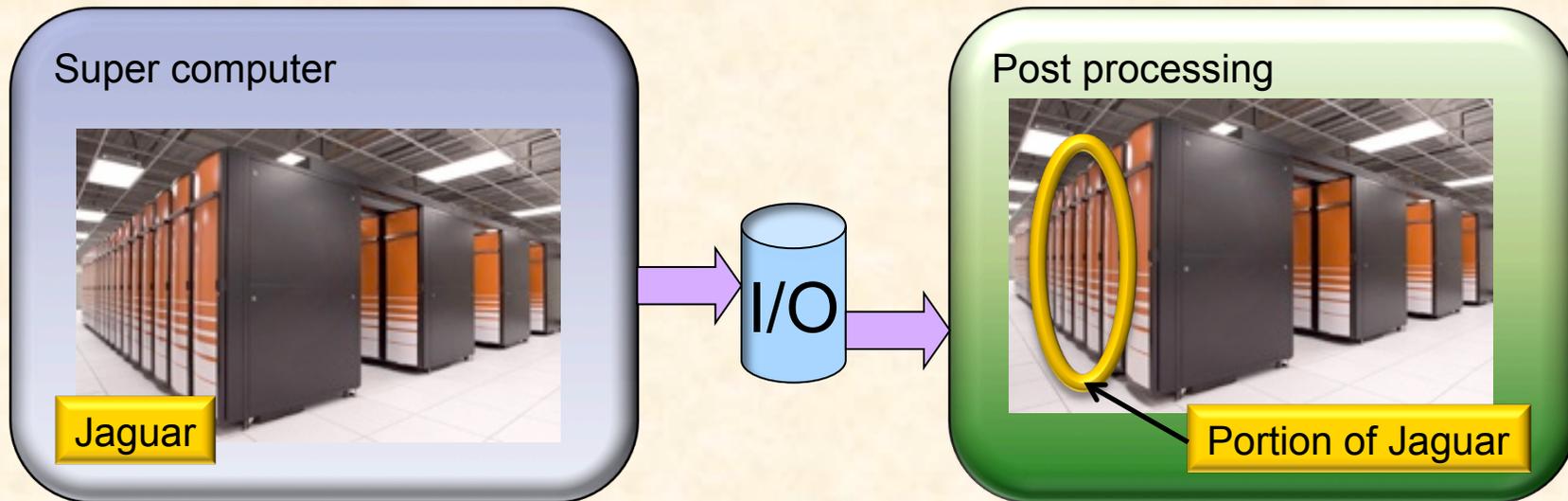
- Challenges: Traditional visualization



# Will not scale

# Production analysis and viz

- Challenges: Shared resource visualization



- Lightweight OS on compute nodes
- Viz apps use a lot of full Linux
- No graphics cards
- Compositing

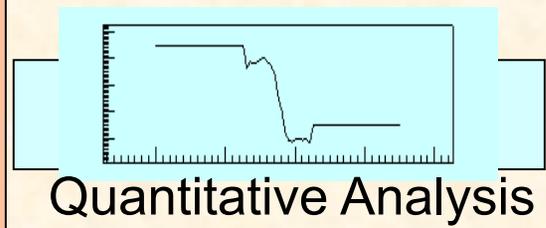
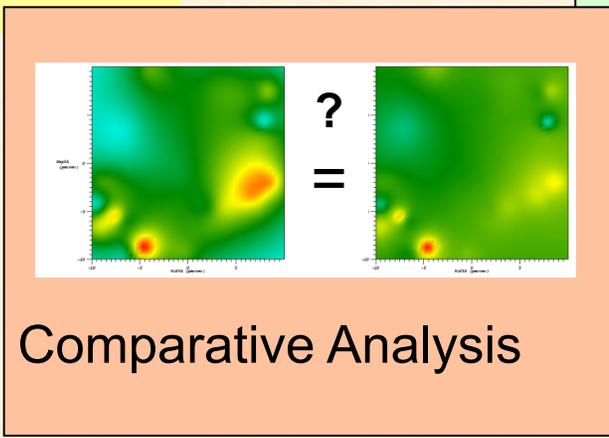
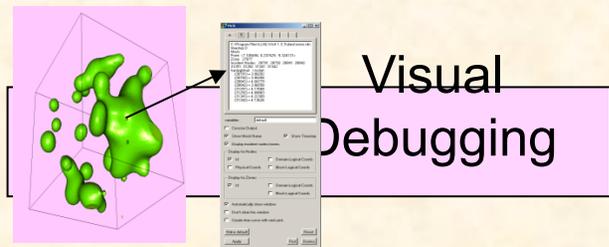
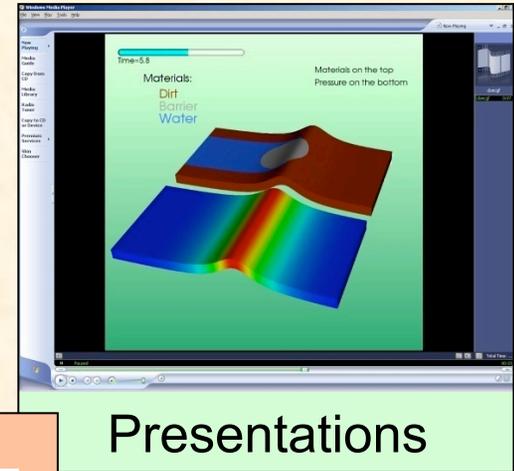
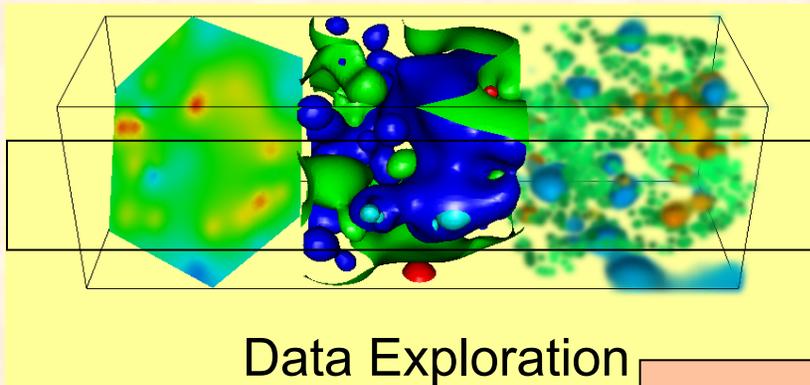
## Visit Overview



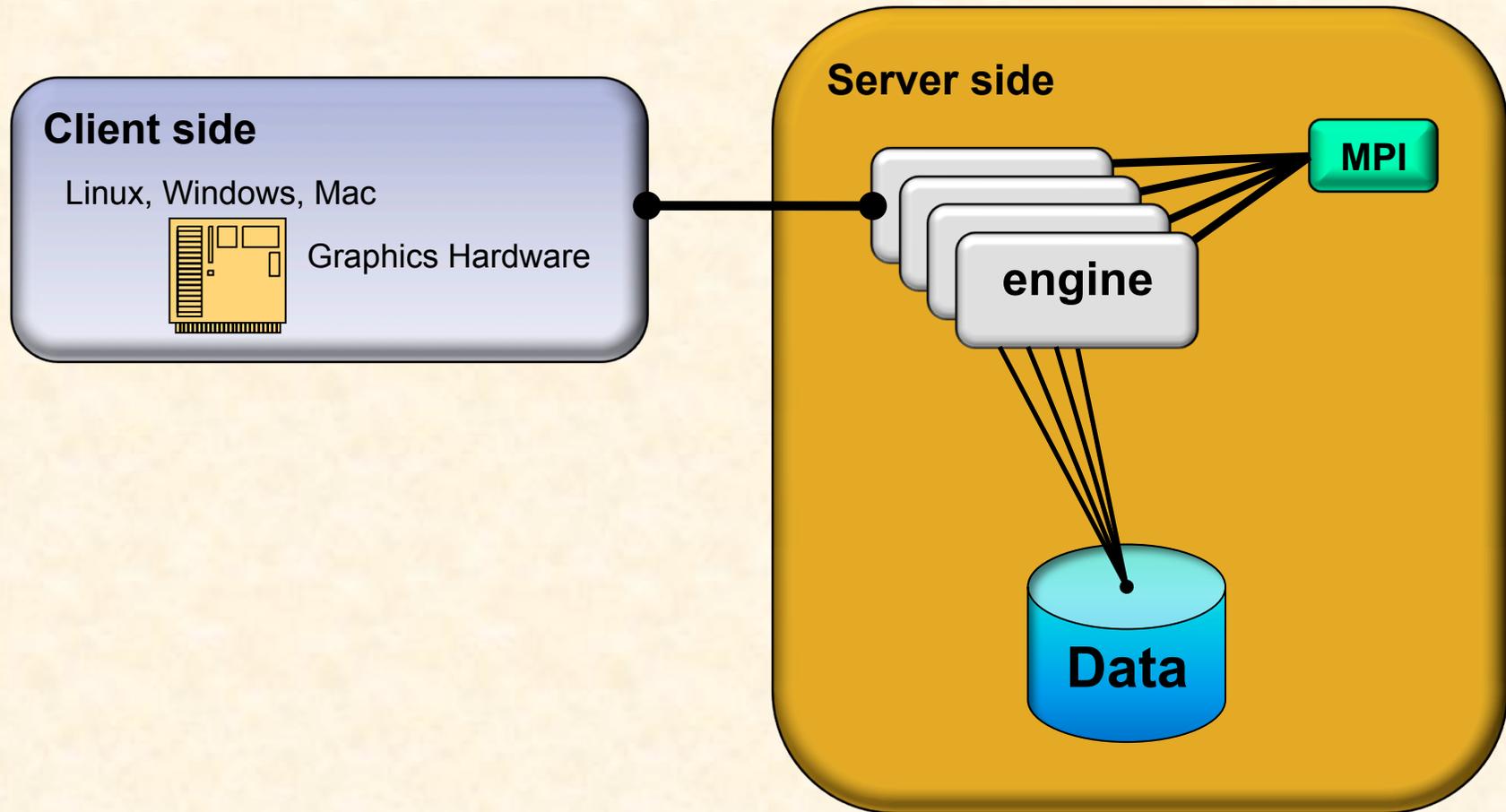
- **Visit is an open source, end user visualization and analysis tool**
  - **Used by: physicists, engineers, code developers, vis experts**
  - **~50 > simulation codes < ~100**
  - **~1000 users at LLNL, ORNL, LBL, others...**
  - **>100K downloads on web**
  - **Developers at LLNL, ORNL, LBL, UC Davis, UUtah.....**

[www.llnl.gov/VisIt](http://www.llnl.gov/VisIt)

# Use cases include...



# VisIt Architecture



# Port to Catamount

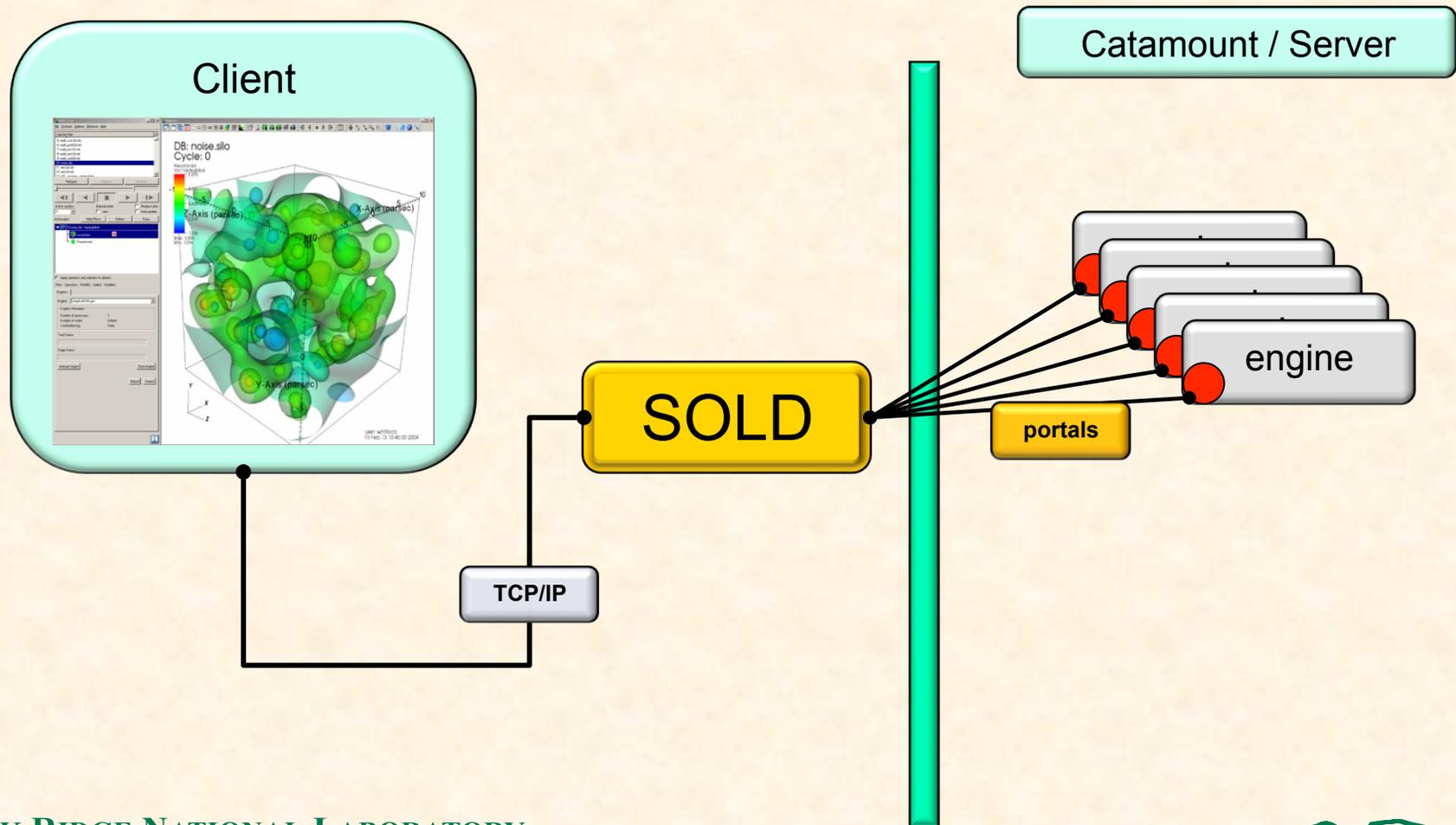
Kevin Thomas, CUG 2007

## Catamount: Light kernel

Issues	Solutions
Static link only	<ul style="list-style-type: none"><li>• Static build of VisIt</li><li>• Limit plugin set</li></ul>
No sockets	<b>S</b> ocket <b>O</b> ff <b>L</b> oad <b>D</b> aemon

# Port to Catamount

Kevin Thomas, CUG 2007



# Port to CLE

CNL: Less restrictive light kernel

Issues	Solutions
Everything on lustre	<ul style="list-style-type: none"><li>• Install VisIt → /lustre/....</li><li>• cp /usr/lib64/* /lustre/....</li></ul>
Shared libs. Limited support	Engine wrapper

# Engine wrapper:

```
int main( int argc, char **argv )
{
    setenv( "LD_LIBRARY_PATH", "/lustre/Vislt/.... ", 1 );

    char *eng = "/lustre/scratch/pugmire/visit/1.8.0/linux-x86_64/bin/engine_par_exe";

    char cmd[100000];
    sprintf( cmd, "%s", eng );

    //Add argv to cmd

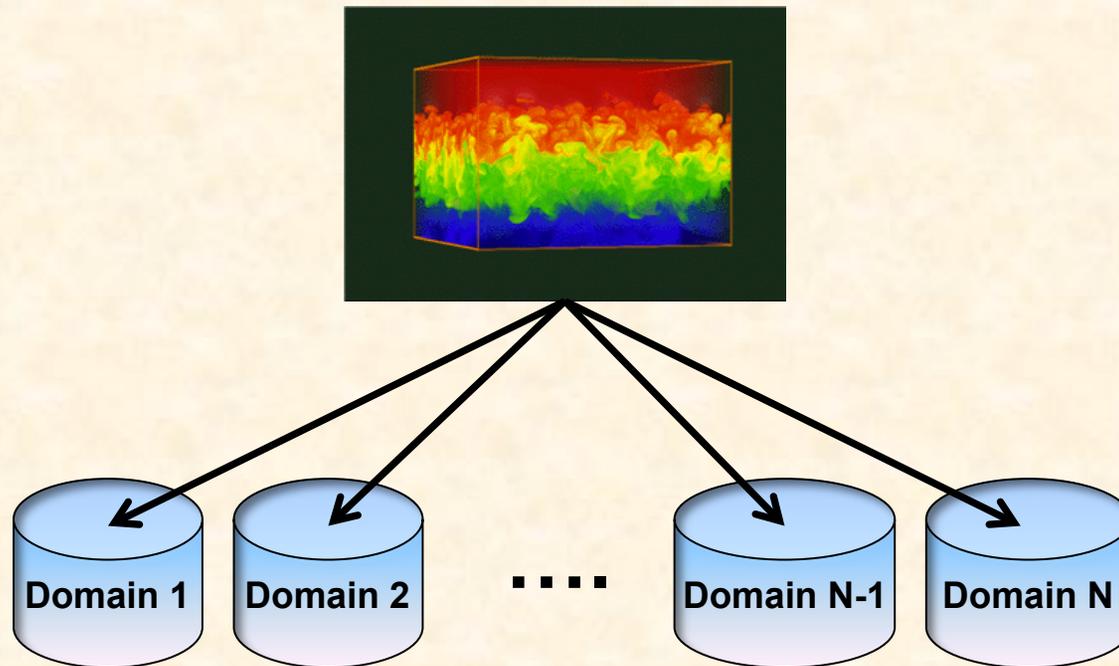
    system( cmd );
    return 0;
}
```

# Port to CLE (XT5)

XT5: Even less restrictive light kernel

Issues	Solutions
Everything on lustre	<ul style="list-style-type: none"><li>• Install VisIt → /lustre/....</li><li>• cp /usr/lib64/* /lustre/....</li></ul>
Shared libs. Limited support	env LD_LIBRARY_PATH=... aprun

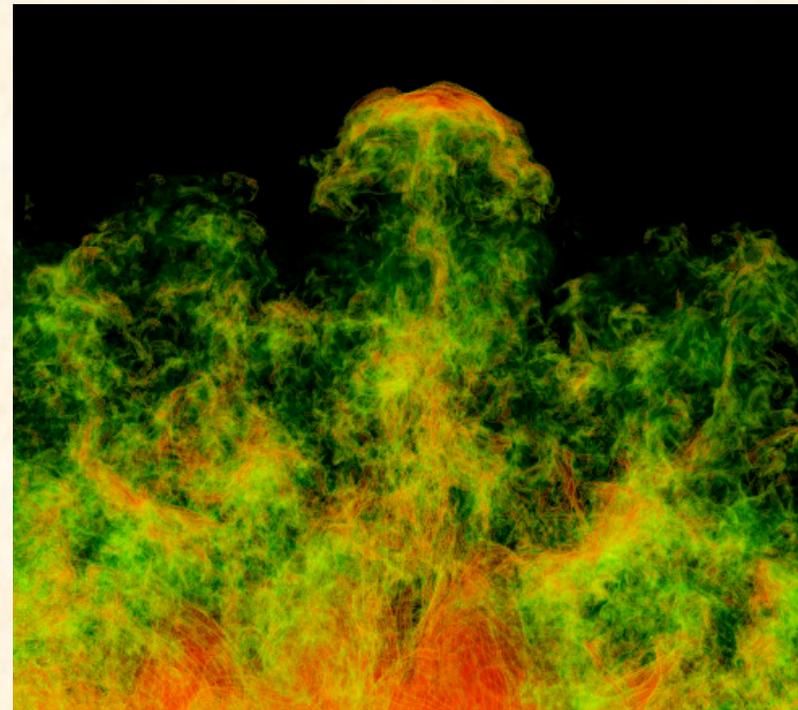
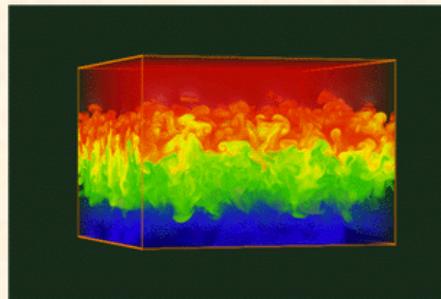
# Test Data



- Load  $N$  domains into memory
- Apply viz algorithms
- Display result

# Results

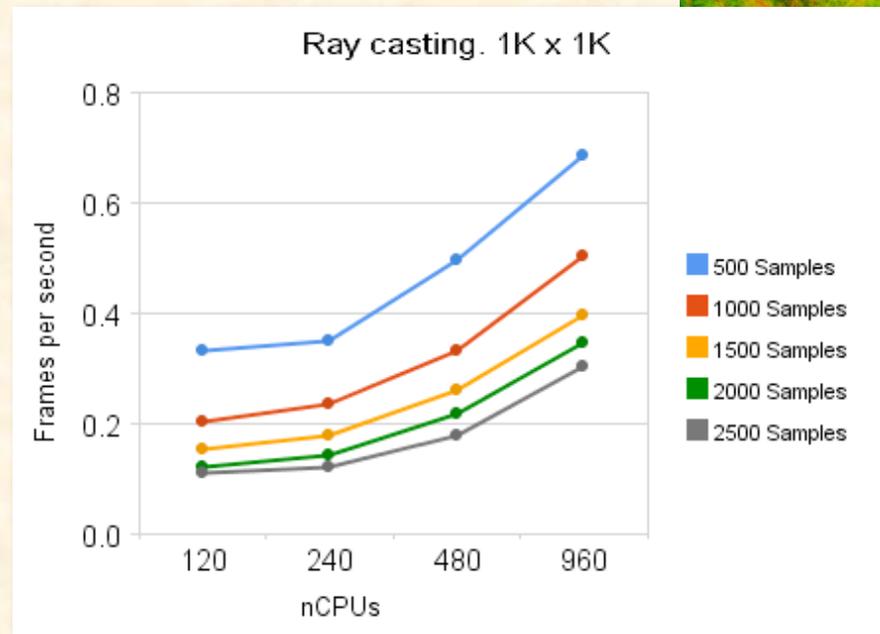
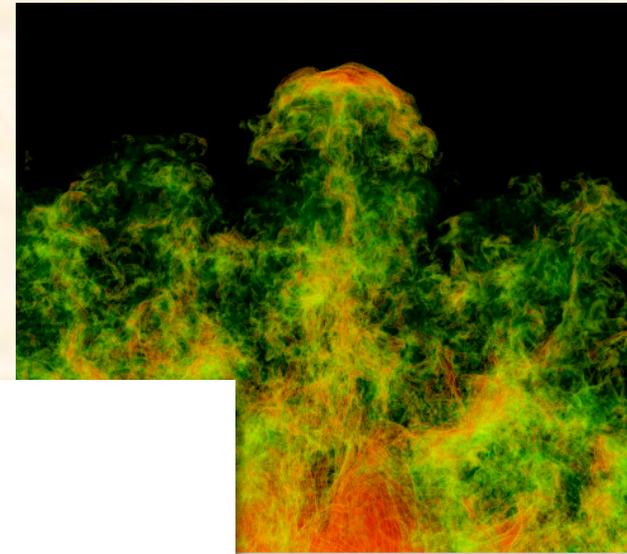
- Interactive exploration of Richtmyer-Meshkoff simulation
- Mesh: 2048x2048x1920
- 960 domains
- Interactive ray casting session



# Results

Interactive exploration.

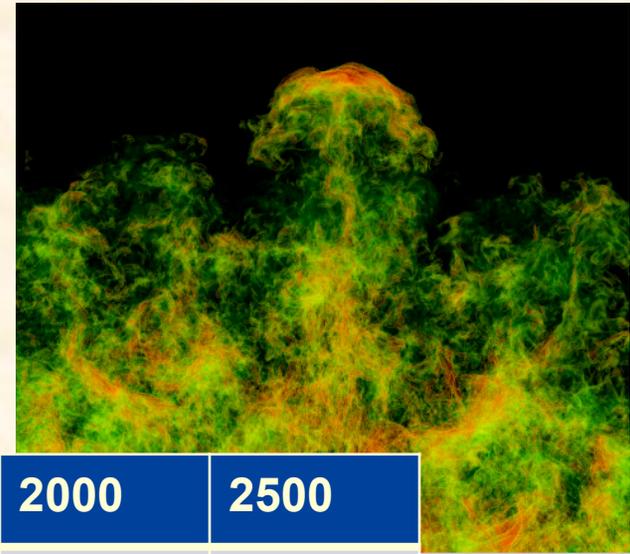
Ray casting 1024x1024 image



# Results

Interactive exploration.

Ray casting 1024x1024 image



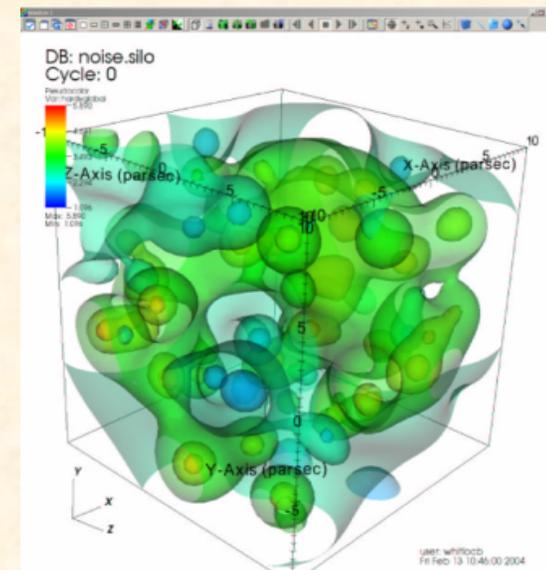
nCPUs	500	1000	1500	2000	2500
120	0.331	0.205	0.153	0.123	0.110
240	0.350	0.236	0.178	0.144	0.121
480	0.497	0.332	0.262	0.218	0.177
960	0.687	0.505	0.396	0.345	0.305

# Results

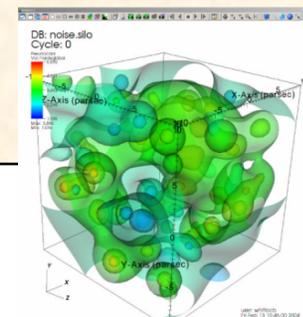
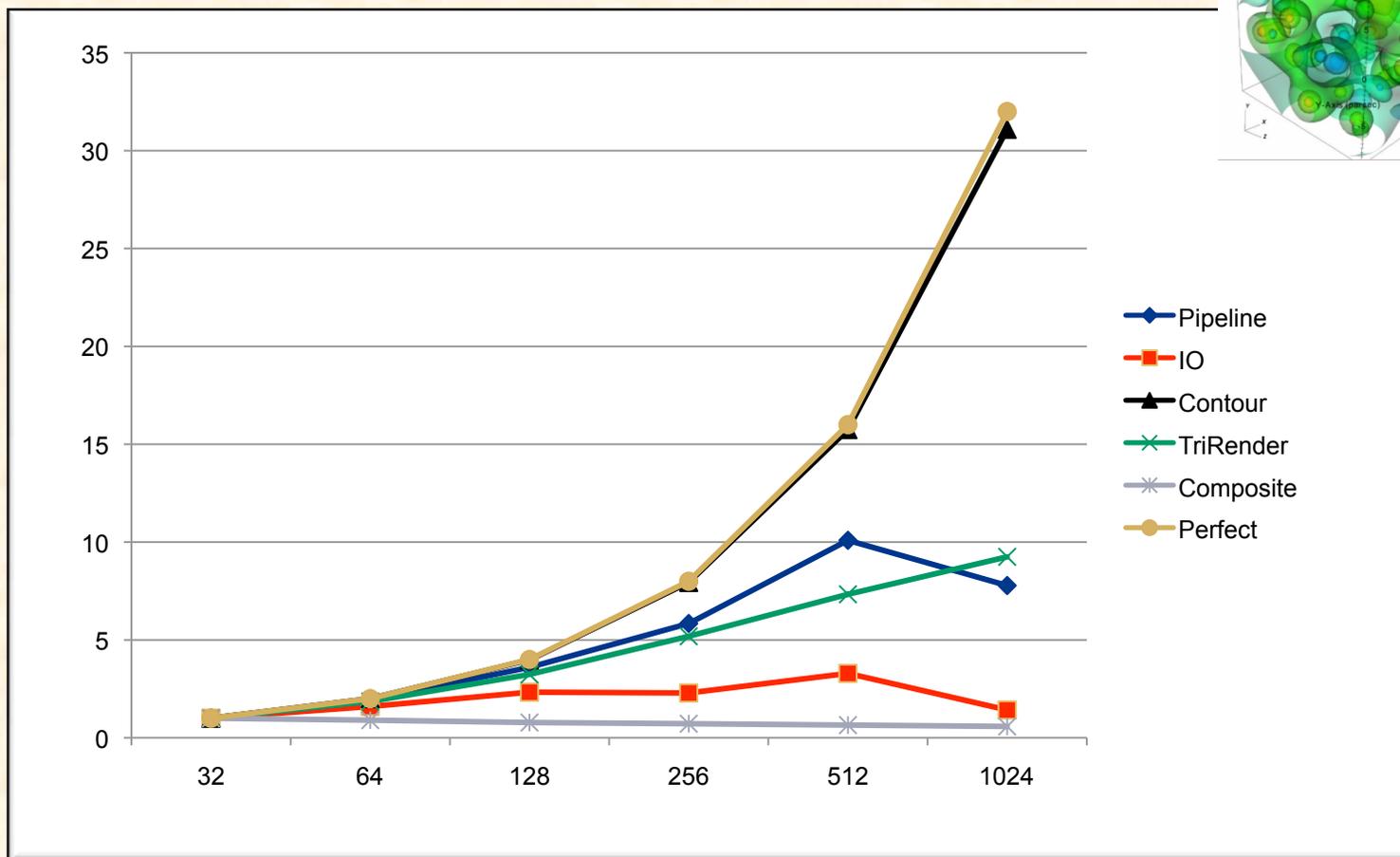
- Isocontour of interpolated *noise* data set

- 1 billion zones, 1000 domains
  - Each file: 4.2 MB
- 10 billion zones, 1000, 2744 domains
  - Each file: 42MB, 16MB

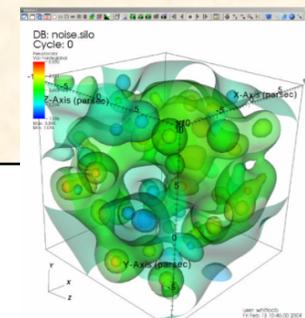
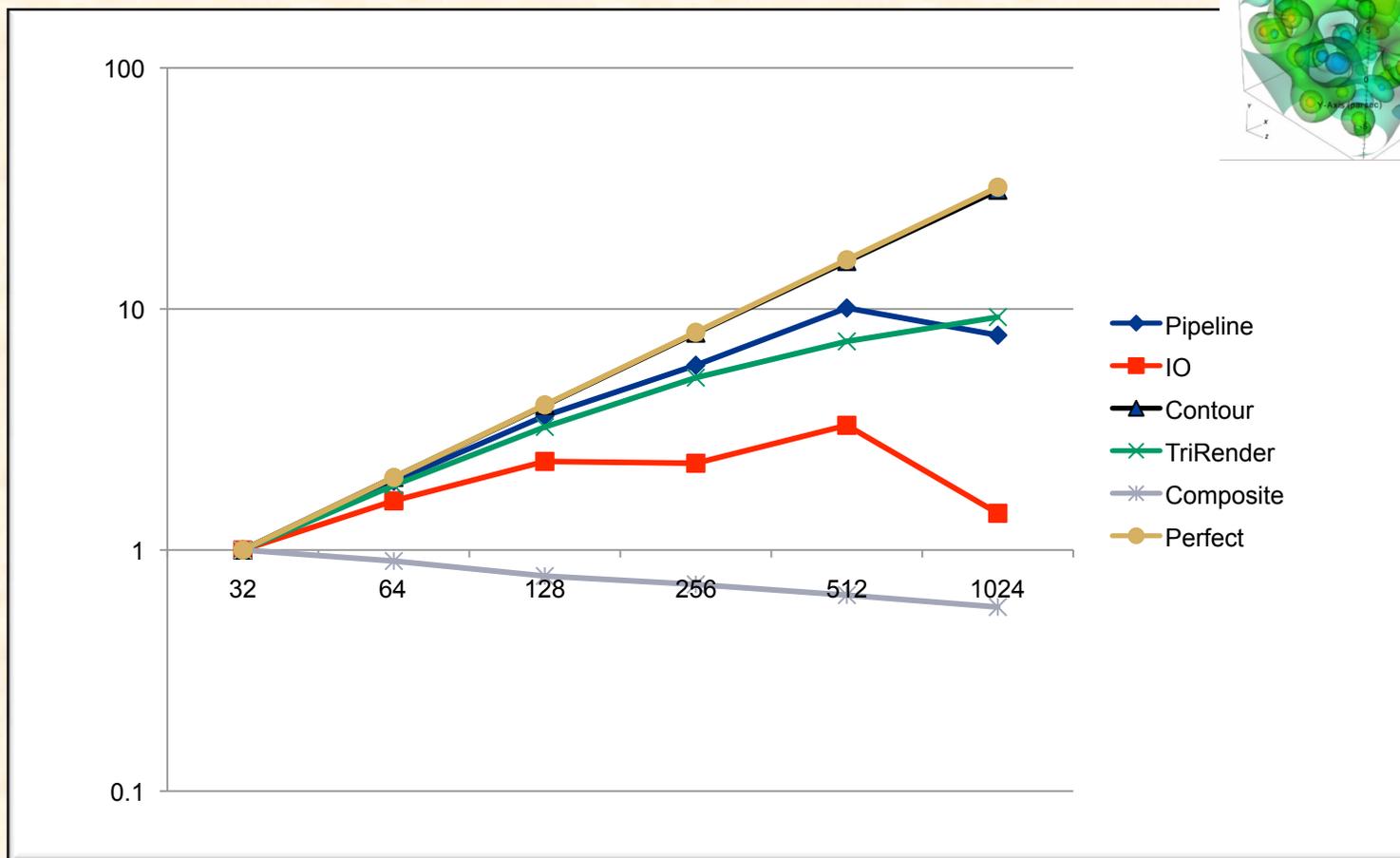
- Load data
- Compute 10 iso contours
- Draw
  1. Render triangles
  2. Composite sub-images (\*\* *Not expected to scale*)
  3. Display result



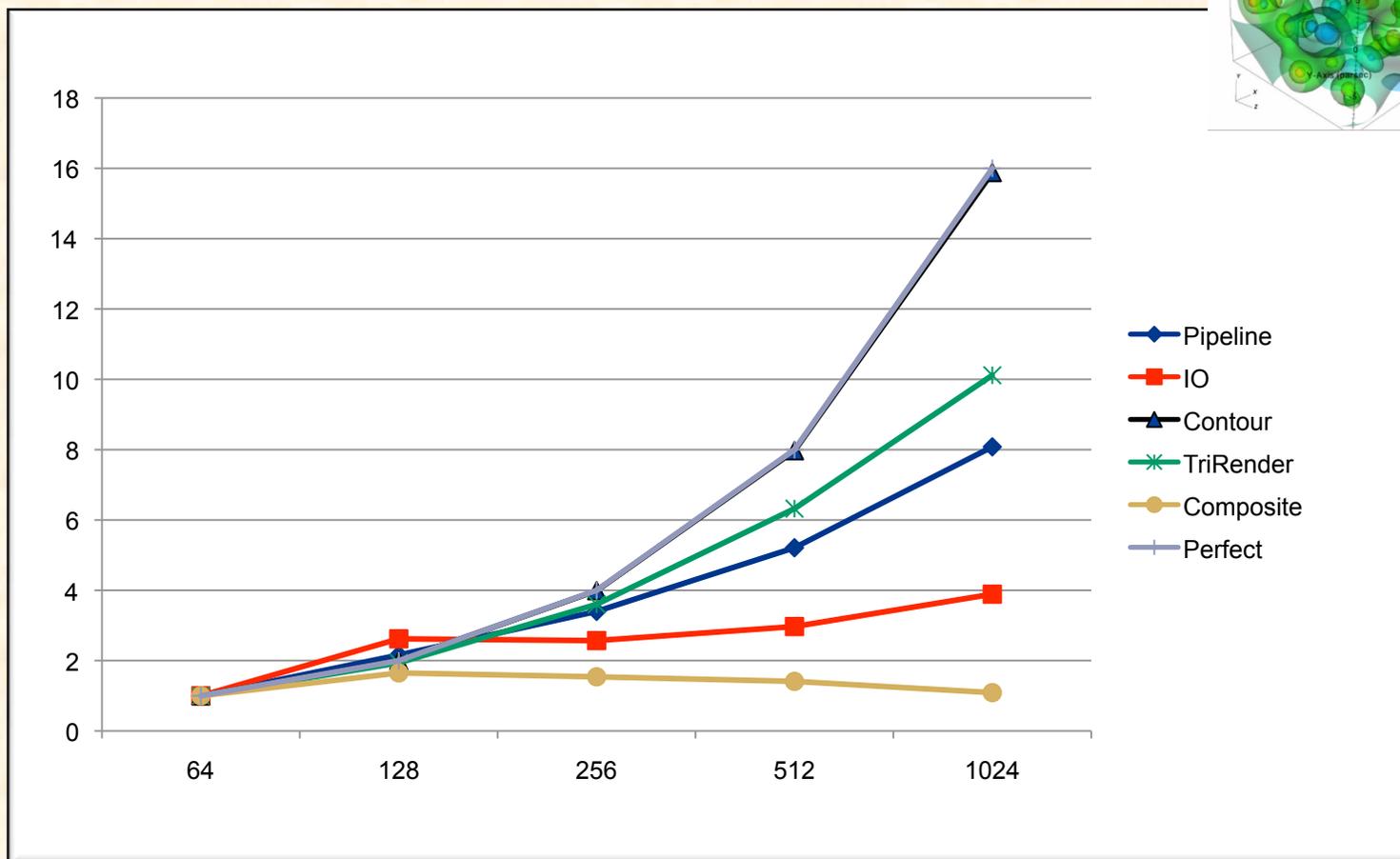
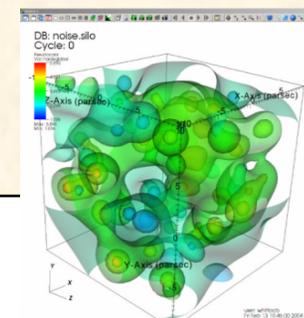
# 1 Billion zones, 1000 domains



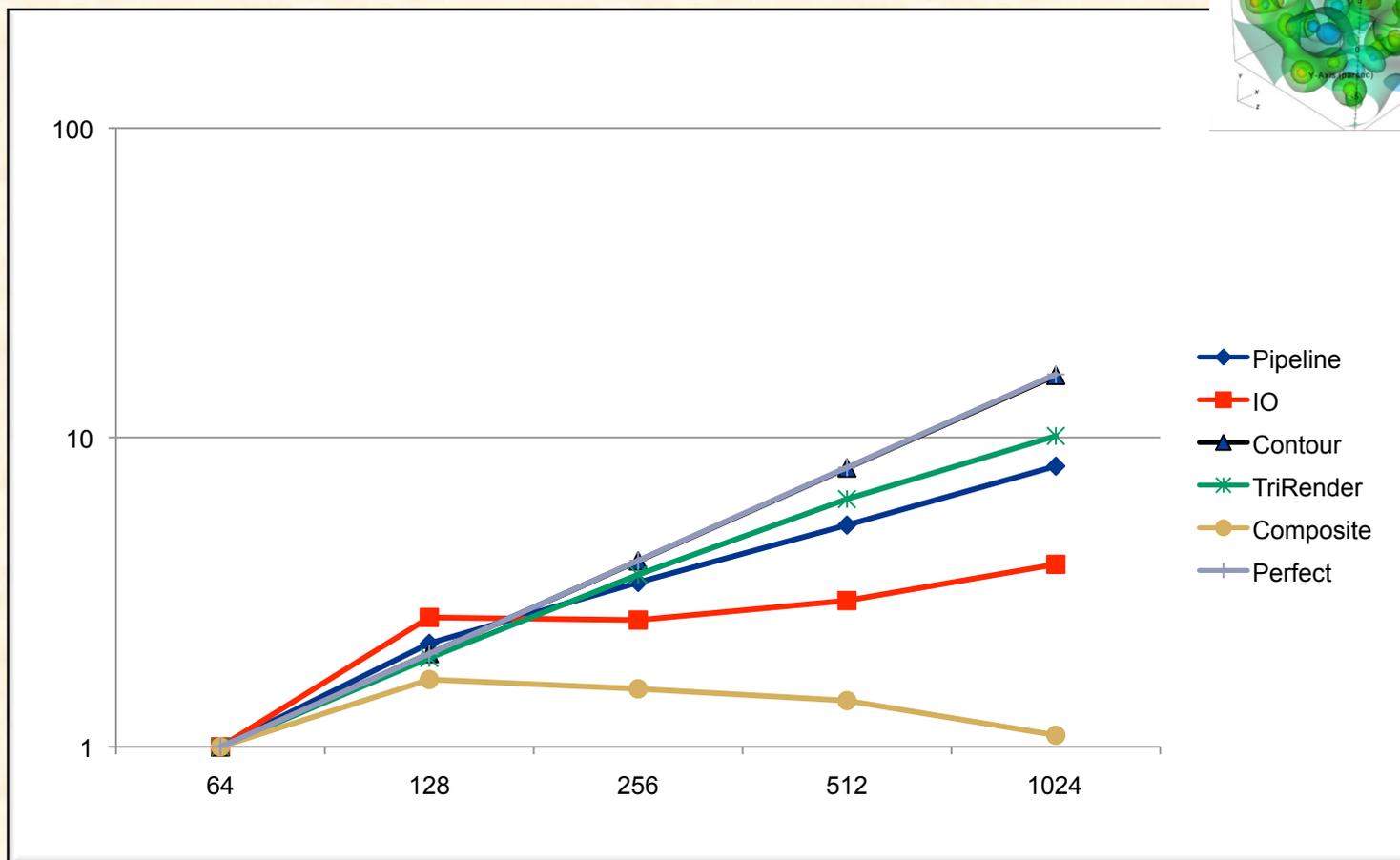
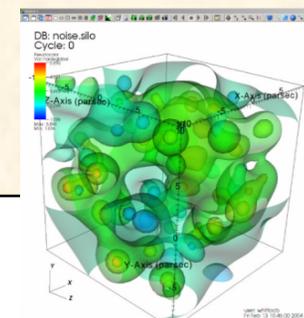
# 1 Billion zones, 1000 domains



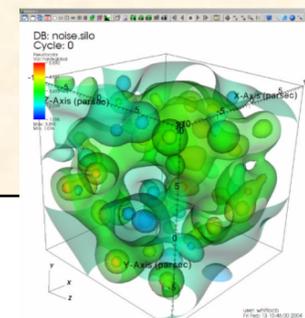
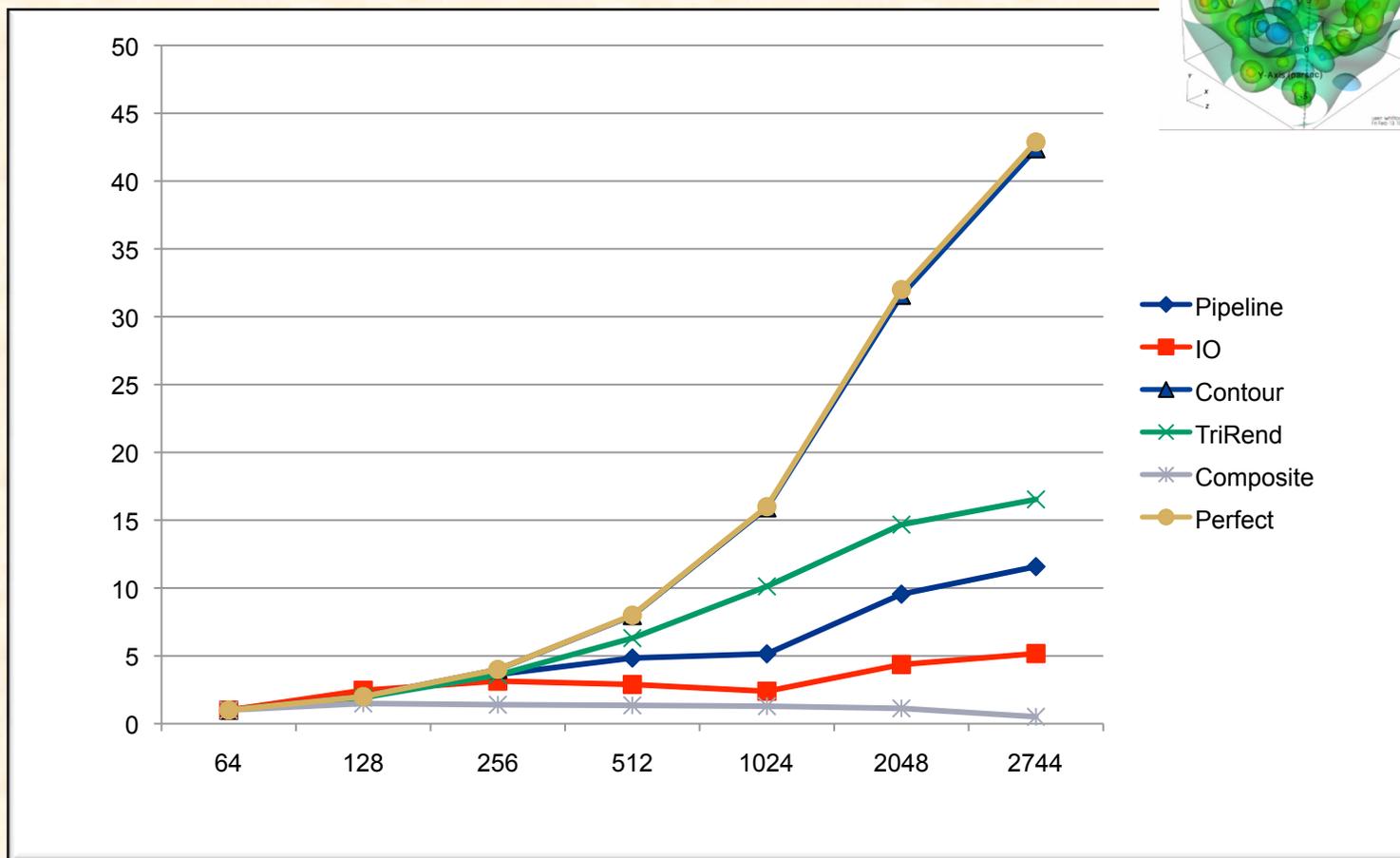
# 10 Billion zones, 1000 domains



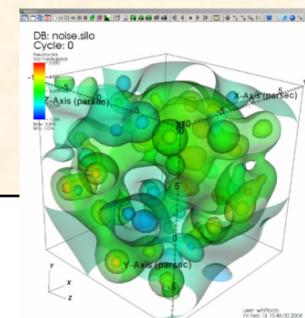
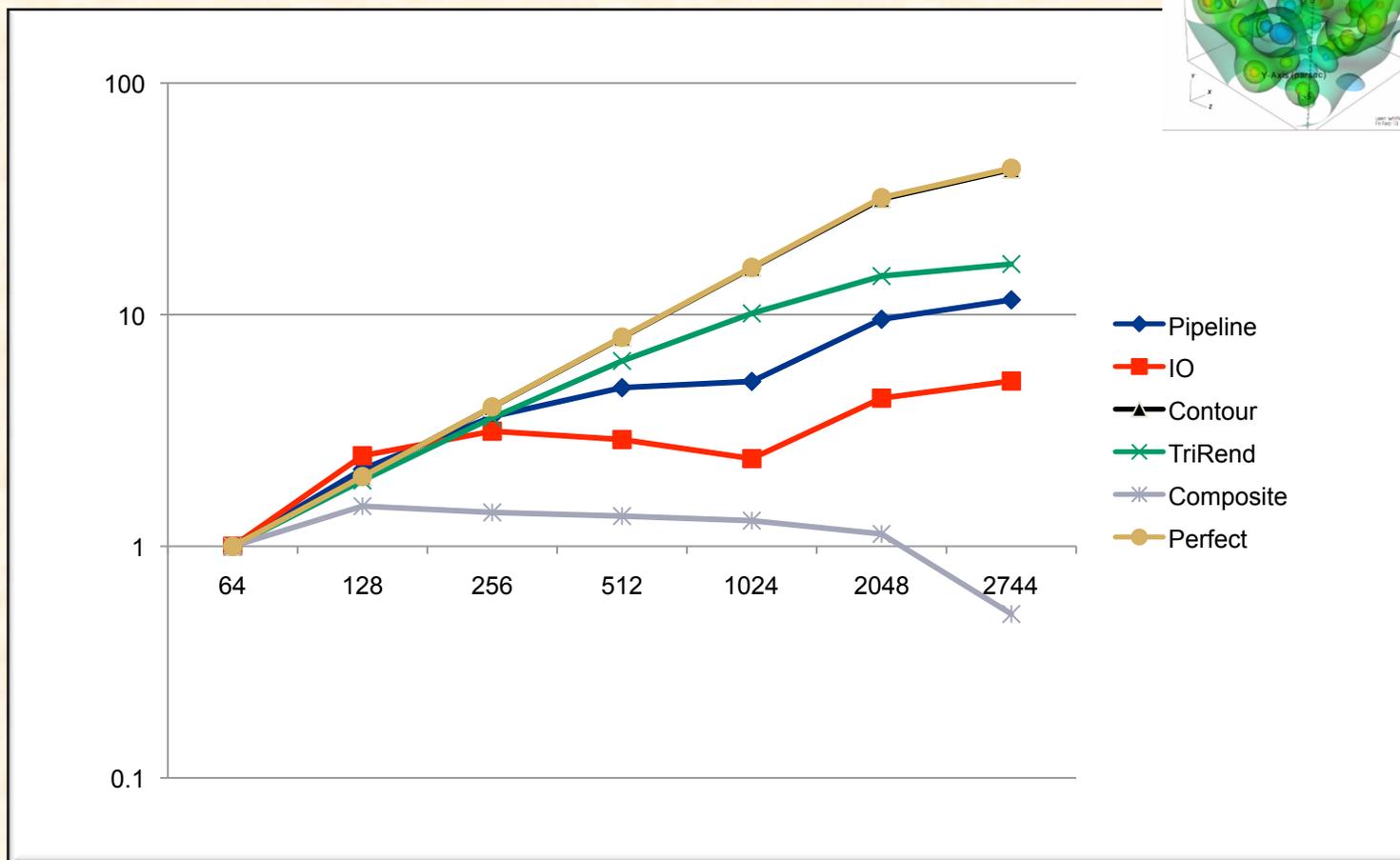
# 10 Billion zones, 1000 domains



# 10 Billion zones, 2744 domains



# 10 Billion zones, 2744 domains



# Conclusions

- Viable way to do analysis and visualization
  - Scaling similar to other known study
- Valuable insight into tool behavior at scale
- More scaling studies needed